AMENDMENTS TO THE CLAIMS

The following listing of claims replaces all prior versions of the claims and all prior listings of the claims in the present application.

1. (currently amended) A Montgomery modular multiplier <u>of a public-key cryptographic system that</u> [[which]] calculates a value congruent to "ABR^{-1}" (mod M) <u>used in the cryptographic system</u>, where A and B are input n-bit numbers, R^{-1} is an inverse number of R modular-multiplied for "mod M", and M is a modulus, the Montgomery modular multiplier comprising:

an A-register storing a bit value a_i (where <u>'i' denotes</u> [[is]] an integer <u>in the range of</u> ~~ranging from~~ 0 to n-1) of the number A, which is smaller than the modulus M;

a B-register storing a bit value b_i of the number B, which is smaller than the modulus M;

an M-register storing a bit value m_i of the modulus M, which is an odd number;

a b_iA calculation logic circuit multiplying the number A by <u>the bit value</u> b_i to obtain b_iA;

a q_i calculation logic circuit solving a Boolean logic equation "s_0 XOR c_0 XOR (b_i AND a_0)", where s_0 is the least significant bit (LSB) of a sum S, c_0 is the LSB of a carry C, b_i is the bit value of the number B, and a_0 is the LSB of the number A, to obtain a bit value q_i;

a q_iM calculation logic circuit multiplying the <u>modulus</u> M by <u>the bit value</u> q_i to obtain q_iM;

a 4-2 compressor [[first]] performing <u>'n'</u> additions on the carry C, the sum S, the b_iA, and the q_iM to obtain interim values <u>and</u> summing the interim values to obtain a result <u>using a</u> [[by]] carry propagation adder in response to a carry propagation adder signal;

an S-register updating a bit value s_i of the sum S and storing the updated bit value; and

a C-register updating a bit value c_i of the carry C and storing the updated bit value.

2. (currently amended) The Montgomery modular multiplier of claim 1, wherein the 4-2 compressor comprises:

a first full adder unit[[,]] summing a bit value b_ia_i of the b_iA, a bit value s_{i+1} of the sum S, and the bit value c_i of the carry C to obtain a carry cA_i and a sum sA_i;

a multiplexer (MUX) unit selectively outputting either a bit value q_im_i [[q_iM_i]] of the q_iM, a [[the]] carry cA_{i-1}, and the sum sA_i or the bit value s_{i+1} of the sum S, the bit value c_i of the carry C, and a [[the]] bit value c_{i-1} of the carry C, in response to the carry propagation adder signal; and

a second full adder unit performing 'n' additions on the bit value q_im_i of q_iM, the carry cA_{i-1}, and the sum sA_i to calculate interim bit values s_i of the sum S and c_i of [[S and]] the carry C, when the carry propagation adder signal is in an inactive state, and [[then]] summing the bit value s_{i+1} of the sum S, the bit value c_i of the carry C, and the bit value c_{i-1} of the carry C to obtain final results of the sum S and the carry C, when the carry propagation adder signal is in an active state.

3. (currently amended) The Montgomery modular multiplier of claim 2 [[1]], wherein a [[the]] carry save adder structure is a 4-input 2-output structure, in which the first and second full adder units operate when the carry propagation adder signal is in the [[an]] inactive state.

4. (currently amended) The Montgomery modular multiplier of claim 2 [[1]], wherein a [[the]] carry propagation adder structure is a 3-input 2-output structure, in which only the second full adder unit operates when the carry propagation adder signal is in the [[an]] active state.

5. (original) The Montgomery modular multiplier of claim 2, wherein the LSB of the carry cA_{i-1} and the LSB of the carry c_{i-1} are in a first logic state.

6. (currently amended) The Montgomery modular multiplier of claim 2, wherein the most significant bit ~~Most Significant Bit~~ (MSB) of the sum S [[s_{i+1}]] is equal to the bit value [[carry]] cA_{n-1} at a clock pulse before the carry propagation adder signal is activated.

7. (currently amended) A method of performing a Montgomery modular multiplication in a Montgomery modular multiplier of a public-key cryptographic system, in which the Montgomery modular multiplier includes registers for storing bit values a_i, b_i, m_i, c_i, and s_i (where 'i' denotes an integer in the range of 0 to [[n*1]] n-1) of a word A, a word B, a modulus M, a carry C, and a sum S, respectively, and calculates a value congruent to "ABR^{-1}" (mod M), where A and B are input n-bit numbers, R^{-1} is an inverse number of R modular-multiplied for "mod M", and M is a modulus, the method comprising:

receiving the number A, the number B, and the modulus M;

multiplying the number A by a bit value b_i to obtain each bit of b_iA;

solving a Boolean logic equation "s_0 XOR c_0 XOR (b_i AND a_0)", where s_0 is the least significant bit (LSB) of a sum S, c_0 is the LSB of a carry C, b_i is the bit value of the number B,

and a_0 is the LSB of the number A, to obtain a bit value q_i ~~(where i denotes an integer in the range of 0 to n*1)~~;

multiplying the ~~number~~ modulus M by the bit value q_i to obtain each bit of q_iM;

performing 'n' additions on the carry C, the sum S, the b_iA, and the q_iM to obtain interim values for each bit of the sum S and the carry C in a carry save adder structure, in response to a carry propagation adder signal; and

summing the interim values to obtain the final results of the sum S and the carry C in a carry propagation adder structure, in response to the carry propagation adder signal.

8. (original) The method of claim 7, wherein the number A is smaller than the modulus M.

9. (original) The method of claim 7, wherein the number B is smaller than the modulus M.

10. (original) The method of claim 7, wherein the modulus M is an odd number.

11. (currently amended) The method of claim 7, wherein the interim values and final results ~~values~~ of the sum S and the interim values and final results ~~values~~ of the carry C are calculated by:

summing a bit value b_ia_i [[b_iA_i]] of the b_iA, a bit value s_{i+1} of the sum S, and a [[the]] bit value c_i of the carry C to obtain a carry cA_i and a sum sA_i;

selectively outputting either a bit value $q_i m_i$ [[$q_i M_i$]] of the $q_i M$, a carry cA_{i-1}, and the sum sA_i or the bit value s_{i+1} of the sum S, the bit value c_i of the carry C, and a bit value c_{i-1} of the carry C, in response to the carry propagation adder signal;

performing 'n' additions on the bit value $q_i m_i$ of the $q_i M$, the carry cA_{i-1}, and the sum sA_i to calculate interim bit values s_i of the sum S and c_i of [[S and]] the carry C, when the carry propagation adder signal is in an inactive state; and

summing the bit value s_{i+1} of the sum S, the bit value c_i of the carry C, and the bit value c_{i-1} of the carry C to obtain final results of the sum S and the carry C, when the carry propagation adder signal is in an active state.

12. (currently amended) The method of claim 7, wherein the carry save adder structure is a 4-input 2-output structure, in which the interim values of the sum S and the carry C are obtained from the $b_i A$ and the $q_i M$ when the carry propagation adder signal is in an inactive state.

13. (currently amended) The method of claim 7, wherein the carry propagation adder structure is a 3-input 2-output structure, in which the final results ~~values~~ of the sum S and the carry C are obtained from the interim values of the sum S and the carry C when the carry propagation adder signal is in an active state.

14. (currently amended) The method of claim 11, wherein the LSB of the carry cA_{i-1} and the LSB of the carry c_{i-1} are in a first logic state.

15. (currently amended) The method of claim 11, wherein the most significant bit (MSB) [[value]] of the sum S [[s_{i+1}]] is equal to the bit value cA_{n-1} at a clock pulse before the carry propagation adder signal is activated.

16. (currently amended) A method of performing radix 2^N Montgomery modular multiplication in a public-key cryptographic system, where N is greater than or equal to 1 [[N>=1]], the method comprising[[;]]:

receiving a multiplicand [[(A)]], a modulus [[(M)]], and a multiplier [[(B)]];

performing carry save addition using first and second full adder units on at least four inputs related to the multiplicand, modulus, and multiplier to generate a result in redundant representation; and

performing carry propagation addition using the second full adder unit to generate a result in normal representation in response to a carry propagation adder signal.

17. (currently amended) A Montgomery modular multiplier of a public-key cryptographic system, comprising:

a multiplicand [[(A)]] register, storing a bit value a_i of a [[the]] number A;

a modulus [[(M)]] register, storing a bit value m_i of ~~the number~~ a modulus M;

a multiplier [[(B)]] register, storing a bit value b_i of a [[the]] number B;

a b_iA calculation logic circuit multiplying the number A by a bit value b_i to obtain each bit of b_iA;

a q_i calculation logic circuit solving a Boolean logic equation "s_0 XOR c_0 XOR (b_i AND a_0)", where s_0 is the least significant bit (LSB) of a sum S, c_0 is the LSB of a carry C, b_i is the bit

value of the number B, and a_0 is the LSB of the number A, to obtain a bit value q_i (where 'i' denotes an integer in the range of 0 to [[n*1]] n-1);

a q_iM calculation logic circuit multiplying the modulus M by the bit value q_i to obtain each bit of q_iM; and

a t-s compressor, wherein t>3[[,]] and s>1, performing 'n' additions on the carry C, the sum S, the b_iA, and the q_iM to obtain interim values for each bit of the sum S and the carry C in a carry save adder structure and summing the interim values to obtain final results of the sum S and the carry C in a carry propagation adder structure, in response to a carry propagation adder signal.

18. (new) A system embodying a Montgomery modular multiplier, the system comprising:

an A-register storing a bit value a_i (where 'i' denotes an integer in the range of 0 to n-1) of an n-bit number A;

a B-register storing a bit value b_i of an n-bit number B;

an M-register storing a bit value m_i of an n-bit modulus M;

a b_iA calculation logic circuit multiplying the number A by the bit value b_i to obtain b_iA;

a q_i calculation logic circuit solving a Boolean logic equation "s_0 XOR c_0 XOR (b_i AND a_0)", where s_0 is the least significant bit (LSB) of a sum S, c_0 is the LSB of a carry C, b_i is the bit value of the number B, and a_0 is the LSB of the number A, to obtain a bit value q_i;

a q_iM calculation logic circuit multiplying the modulus M by the bit value q_i to obtain q_iM;

a compressor performing 'n' additions on the carry C, the sum S, the b_iA, and the q_iM to obtain interim values and summing the interim values to obtain a result using a carry propagation adder in response to a carry propagation adder signal;

an S-register updating a bit value s_i of the sum S and storing the updated bit value; and

a C-register updating a bit value c_i of the carry C and storing the updated bit value;

wherein given that the number A is smaller than the modulus M, the number B is smaller than the modulus M, the modulus M is odd, and R^{-1} is an inverse number of R modular-multiplied for "mod M", the system calculates a value congruent to "ABR^{-1}" (mod M).

19. (new) The system of claim 18, wherein the system comprises a public-key cryptographic system.

20. (new) The system of claim 18, wherein the system is a public-key cryptographic system.

21. (new) The system of claim 18, wherein the value congruent to "ABR^{-1}" (mod M) is used in a public-key cryptographic system.

22. (new) The system of claim 18, wherein the value congruent to "ABR^{-1}" (mod M) is used in the system as a cryptographic key.